LAPIN AMK
Lapland University of Applied Sciences

AUTOMAATIOTESTAUS MOBIILILAITTEILLA

Jokelainen Tuomas

Opinnäytetyö

Tieto- ja viestintätekniikka
Insinööri (AMK)

VUOSI 2020

LAPIN AMK
Lapland University of Applied Sciences

Opinnäytetyön tiivistelmä

Tieto- ja viestintätekniikka
Insinööri(AMK)

Tekijä	Tuomas Jokelainen	Vuosi	2020
Ohjaaja	Aku Kesti		
Toimeksiantaja	Mtech Digital Solutions Oy		
Työn nimi	Automaatiotestaus mobiililaitteilla		
Sivu- ja liitesivumäärä	23 + 2		

Tämän opinnäytetyön ensimmäisenä tavoitteena oli selvittää, onko automaatiotestaus mahdollista eri mobiililaitteilla ja mitä työkaluja se vaatii. Toiseksi haluttiin testata, onko mahdollista kirjoittaa yksi testikoodi niin, että se toimisi kaikilla testilaitteilla. Työssä halutaan myös selvittää, onko mobiililaitteiden testausta mahdollista tehdä sisäverkon yli kahden tietokoneen välillä.

Opinnäytetyön toimeksiantajana toimi Mtech Digital Solutions Oy. Olen itse töissä yrityksen Rovaniemen toimipisteessä ohjelmistotestaajana, minkä vuoksi verkkosivujen automaatiotestaus on minulle tuttua. Alalla mobiililaitteiden automaatiotestausta on tehty aiemminkin, mutta meidän yksikössämme se ei ole ollut käytössä.

Aluksi etsin tietoa, mitä työkaluja mobiililaitteiden automaatiotestauksessa on käytetty muualla. Lopulta päädyin valitsemaan työkaluiksi Robot Frameworkin ja Appiumin. Tutkimusmenetelmäksi valitsin toiminnallisen tutkimuksen, jossa testasin mobiililaitteen omaa sovellusta (laskin) sekä verkkoselainpohjaista sovellusta (MtechMap). Testilaitteina minulla oli kaksi matkapuhelinta ja kaksi tablettitietokonetta. Testausta rajoitti esimerkiksi se, että minulla ei ollut käytettävissä laitteita ja ohjelmistoja iPhonelle ja iPadille.

Johtopäätöksenä voidaan todeta, että verkkoselainpohjaiselle sovellukselle tehtävä automaatiotestaus ei ole mahdollista usealla mobiililaitteella käyttäen vain yhtä testikoodia. Mobiililaitteiden alkuperäissovelluksen testaamisessa huomattiin, että sovellukset ovat liian erilaisia, mikä vaati paljon koodimuutoksia. Automaatiotestit saatiin tehtyä onnistuneesti jokaiselle mobiililaitteelle erikseen, mutta yhden testikoodin käyttäminen eri laitteille todettiin mahdottomaksi. Sisäverkon yli testaus kahden tietokoneen välillä oli yksinkertaista ja toimi helposti valituilla ohjelmistoilla.

Avainsanat Appium, automaatiotestaus, Robot Framework

LAPIN AMK
Lapland University of Applied Sciences

Abstract of Thesis

Degree Programme in Information
and Communication Technology
Bachelor of Engineering

Author	Tuomas Jokelainen	Year	2020
Supervisor	Aku Kesti		
Commissioned by	Mtech Digital Solutions Ltd		
Subject of thesis	Automation Testing for Mobile Devices		
Number of pages	23 + 2		

In this thesis, the main objective was to find out if it was possible to make one test code so that it worked on all the selected mobile devices. In addition, it was tested if it was possible to make automation tests between two computers in the same network.

Automation tests were created for a web browser-based application (MtechMap) and a mobile devices' application (calculator). They were tested on four different mobile devices. Robot Framework and Appium were selected as the automation testing tools as they were proven to be a working solution for the task.

As a conclusion, automation testing on multiple mobile devices with one code was not possible for neither the web browser-based application nor the mobile devices' application. It was realized that the applications were too different from each other, which required many changes in the test code. However, the automation tests were successful on individual devices after adjusting the test code on a device basis. Testing between two computers in the same network was simple and worked well with the chosen software.

Key words Appium, automation testing, Robot Framework

SISÄLLYS

ALKUSANAT

Haluaisin kiittää kaikkia työkavereitani Mtech Digital Solutions Oy:llä, sillä he ovat jaksaneet muistuttaa minua opinnäytetyön teosta ja antaneet aikaa opinnäytetyön tekemiselle.

Iso kiitos myös vaimolleni, joka on jaksanut tsempata ja auttaa minua opinnäytetyön teossa. Lisäksi haluan kiittää vanhempiani ja sisaruksiani siitä, että ovat muistuttaneet opinnäytetyöstä kyllästymiseen asti.

KÄYTETYT MERKIT JA LYHENTEET

RPA (robotic process automation)	automaatiorobotiikka
SDK (software development kit)	ohjelmistokehitystyökalu

1 JOHDANTO

Opinnäytetyön toimeksiantaja on Mtech Digital Solutions Oy. Yrityksellä on yhteensä neljä toimipistettä: kaksi Vantaalla, yksi Kauhajoella ja yksi Rovaniemellä. Yrityksen päätoimipiste on Vantaan Jokiniemessä. Yritys on perustettu vuonna 1986 nimellä Maatalouden Laskentakeskus Oy, ja sillä on vahva historia verkkopalvelujen kehittämisessä maatalousalalla. Palvelutarjonnan laajentuessa yritys vaihtoi nimensä Mtech Digital Solutions Oy:ksi vuonna 2016, ja nykyään asiakkaita on monilta eri toimialoilta. (Mtech Digital Solutions Oy 2020.)

Tässä opinnäytetyössä haetaan mahdollisuutta lisätä Mtech Digital Solutions Oy:n Rovaniemen toimipisteen automaatiotestaus suoraan mobiililaitteisiin muun automaatiotestauksen lisäksi. Opinnäytetyö vaikuttaa suoraan omaan työhöni ohjelmistokehittäjänä/ohjelmistotestaajana, ja tästä syystä johtuen aihe on itselleni erittäin mielenkiintoinen.

Opinnäytetyön tarkoituksena on vähentää manuaalisten testien suunnittelua ja niiden auki kirjoittamista. Tämä lisää automaatiotestikoodien kirjoitusta ja niiden ylläpitoa, mikä pidemmällä aikavälillä kuitenkin säästää työaikaa. Useimmiten testauksessa kirjoitetaan käyttötapausdokumentti, jossa kuvataan kaikki testin vaiheet. Liitteenä on esimerkki kirjoitetusta manuaalitestin suunnitelmasta (Liite 1). Automaatiotestauksessa samanlaista dokumenttia ei tarvita, sillä tiedot löytyvät testin lokitiedostosta.

Työssä käydään läpi automaatiotestauksen vaiheet sekä tulokset siitä, miten robotiikka toimii eri mobiililaitteilla. Testauksessa käytetään tässä vaiheessa vain laitteiden omia sovelluksia ja verkkoselainta (Google Chrome).

2 TYÖN VAIHEET

2.1 Tavoite

Tavoitteena opinnäytetyössä on saada ensiksi todettua, toimiiko automaatiotestaus mobiililaitteilla, kun käytetään Robot Frameworkia, sen Appium-kirjastoa sekä Appiumia. Opinnäytetyössä halutaan myös selvittää, onko mahdollista kirjoittaa vain yksi testikoodi niin, että se toimii kaikilla laitteilla. Opinnäytetyössä testataan, onko eri Android-versioiden tai yleisesti ottaen eri laitteiden välillä eroavaisuuksia.

Seuraavaksi on tarkoitus testata, pystyykö automaatiotestejä tekemään sisäverkon yli kahden tietokoneen välillä käyttämällä samoja työkaluja. Tämä mahdollistaisi sen, että mobiililaitteet vastaavat sisäverkon yli ilman suoraa yhteyttä omaan tietokoneeseen.

Tarkoituksena on myös testata, kuinka helposti mobiililaitteille voi tehdä automaatiotestejä, jotta saataisiin nopea hyöty tämänhetkisistä testauslaitteista. Useimmiten automaatiotestit ovat regressiotestejä, jotka kertovat, toimiiko testattava sovellus halutulla tavalla. Automaatioregressiotesti tehdään yleensä silloin, kun sovelluksen koodiin lisätään uusia ominaisuuksia tai toiminnollisuuksia. Mikäli regressiotestit menevät onnistuneesti läpi, voidaan päätellä, että sovellus toimii niiltä osin oikealla tavalla. (Atrsoft 2018.)

2.2 Mobiililaitteiden automaatiotestauksen perusteet

Automaatiotestausta tehdään aina manuaalisen testauksen rinnalle. Yleisimmät automatisoinnin kohteet ovat sovelluksen osat, joiden kuuluu toimia tietyllä tavalla, jolloin niille on helppo tehdä automaatiotesti. Tämä helpottaa testaajan työtä ja antaa merkittäviä työaikasäästöjä pidemmällä aikavälillä manuaaliseen testaukseen verrattuna. (Atrsoft 2018.)

Testausrobotiikkaa on tehty aikaisemminkin mobiililaitteilla. Yhtenä esimerkkinä on Qubilean toimitusjohtaja Janne Lepistön artikkeli Testiautomaation maailma: Robot Framework ja Appium. Artikkelissa hän kertoo hieman yrityksen omasta projektistaan, jossa he ovat tehneet mobiililaitetestausta käyttäen Appiumia ja

Robot Frameworkia. Artikkeli kertoo myös hieman taustaa, miten se tehdään ja mitä siihen tarvitaan. (Lepistö, J. 2016.)

Itse olen omassa työssäni käyttänyt Robot Frameworkia automaatiotestien tekoon verkkosivuille, mikä helpotti koodin kirjoittamista tässä opinnäytetyössä. Robot Frameworkilla tehdään avainsana-pohjaista (keyword driven) koodia, josta esimerkki alla (Kuviot 1 ja 2).

```
Calculator base test Honor 7A
  open calculator Honor
  Minus Honor
  Plus Honor
  Divide Honor
  Times Honor
  Quit Application
```

Kuvio 1. Avainsana-pohjainen automaatiotesti

Avainsana-pohjainen koodi toimii loogisesti, suorittaen aina yhden avainsanan kerrallaan ylhäältä alaspäin (Kuvio 1). Kuviossa 2 nähdään kahden avainsanan (open calculator Honor ja Minus Honor) vaiheet.

```
*** Keywords ***
open calculator Honor
  Open Application  ${REMOTE_URL}  platformName=${PLATFORM_NAME}

Minus Honor
  Click Element  com.android.calculator2:id/digit_6
  Click Element  com.android.calculator2:id/dec_point
  Click Element  com.android.calculator2:id/digit_9
  Click Element  com.android.calculator2:id/op_sub
  Click Element  com.android.calculator2:id/digit_3
  Capture Page Screenshot  MinusHonor.png
  Click Element  com.android.calculator2:id/eq
  Click Element  com.android.calculator2:id/op_clr
```

Kuvio 2. Avainsanan vaiheet

Opinnäytetyön perustana on käytetty Appium-kirjaston (AppiumLibrary 2020) ja Robot Frameworkin (RobotFramework 2020) kirjastojen tietoja, ohjedokumentaatioita sekä suoria koodiesimerkkejä. Pohjatietona toimivat myös aiemmat kokemukseni Robot Frameworkilla tehdyistä automaatiotesteistä verkkosivuille.

Opinnäytetyössä käytettävissä olevista työkaluista Appium on helppokäyttöisin työkalu laitepoolin jakoon verkon yli. Tavoitteena on, että Appiumia käyttämällä

verkon yli testaamiseen ei tarvita erillisiä säätöjä, jos projektia jatketaan isompaan käyttötarkoitukseen.

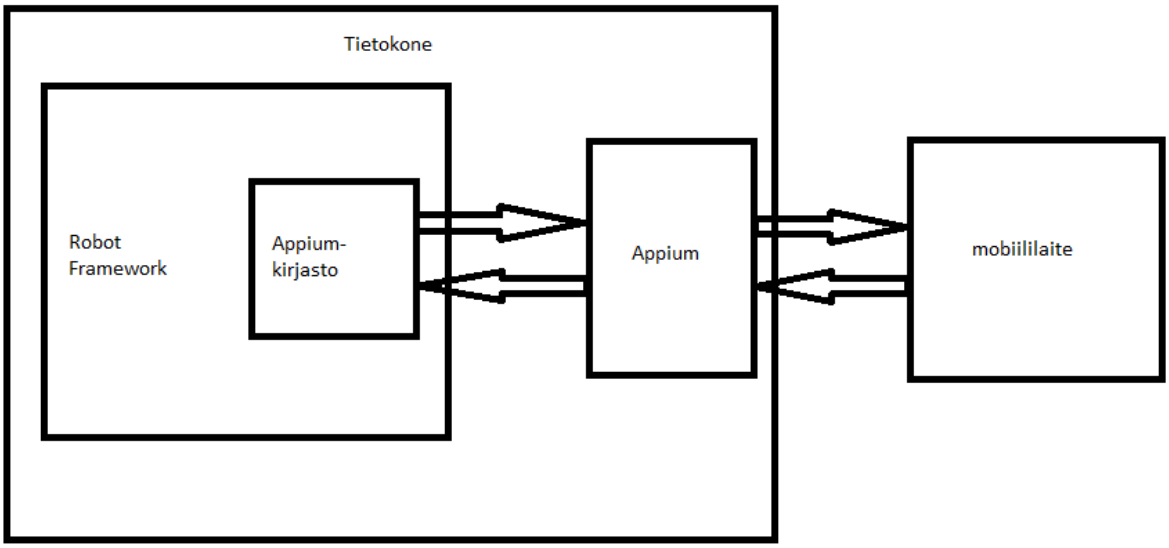
2.3 Valitut työkalut

Robot Framework on avoimen lähdekoodin työkalu, joten siinä ei ole lisensointi- tai muita kustannuksia. Sitä käytetään ohjelmistotestauksessa ja ohjelmistorobotiikassa (RPA lyhenne). Robot Frameworkia on mahdollista laajentaa erilaisilla kirjastoilla ja työkaluilla, joiden koodi on kirjoitettu joko Pythonilla tai Javalla. Robot Frameworkille kirjoitettua koodia pystyy lukemaan kuka vain, sillä siinä on yksinkertainen syntaksi ja helppolukuiset avainsanat. (RobotFramework 2020.)

Appium on sovellus, joka ottaa yhteyden mobiililaitteeseen tai mobiiliemulaatioon mahdollistaakseen yhteyden tietokoneen ja mobiililaitteen välillä verkon yli. Appium toimii tiedonvälittäjänä laitteiden välillä. Appiumilla voi hallita mobiililaitetta niin, että näppäinpolut ovat helposti löydettävissä. (Appium 2020.)

Toimiakseen oikein Appium tarvitsee myös NodeJS-kirjastoja sekä SDK:t(lyhenne). NodeJS on JavaScriptin ajonaikainen ympäristö, jota käytetään Appiumin ajossa. NodeJS:llä pitää olla erikseen asennettuna npm:llä (node package manager) tiettyjä kirjastoja, esimerkiksi adb (Android debug bridge) ja Maven.

Appium-kirjasto on Robot Frameworkin käyttäjien tekemä kirjasto, jolla voidaan käyttää Appiumia suoraan Robot Frameworkin kautta. Appium-kirjaston komentojen kautta on mahdollista kutsua Appiumia Robot Frameworkille. (Appium-Library 2020.) Alla olevassa kuviossa 3 on selitetty yksinkertaistettuna, miten eri sovellukset keskustelevat keskenään.



Kuvio 3. Tiedonkulku osien välillä

3 LAITTEISTO JA TESTIEN AJOT

3.1 Testien valitseminen

Opinnäytetyössä päädyttiin tekemään kaksi erilaista automaatiotestiä. Mtech Digital Solutions Oy tekee pääsääntöisesti verkkoselainpohjaisia sovelluksia, joten toinen testi tehtiin nimenomaan verkkoselainsovellukselle. Sovellukseksi valittiin yrityksen oma sovellus, MtechMap. Automaatiotestissä testattiin sisäänkirjautumista sovellukseen, minkä jälkeen tarkistettiin, onko sisäänkirjautumista seuraava näkymä oikeanlainen.

Toisessa automaatiotestissä päätettiin testata mobiililaitteen perussovellusta, laskinta, sillä sen perusominaisuudet eri laitteilla ovat samat. Laskintestiin tehtiin yksinkertaiset plus-, miinus-, kerto- ja jakolaskut, sillä näiden tuloksia on helppo verrata keskenään.

Testikoodin kirjoittamiseen käytettiin pohjana Appium-kirjaston tietoja, josta löytyivät esimerkit kaikille käytettäville ominaisuuksille (AppiumLibrary 2020). Tämän avulla oli helppo kirjoittaa ensimmäinen testikoodi.

3.2 Testaus eri laitteilla

Testilaitteiksi valittiin kolme matkapuhelinta ja kolme tablettia. Oletuslaitteeksi näistä valittiin Honor 7A -matkapuhelin, jonka tuloksiin verrattiin muiden testilaitteiden tuloksia.

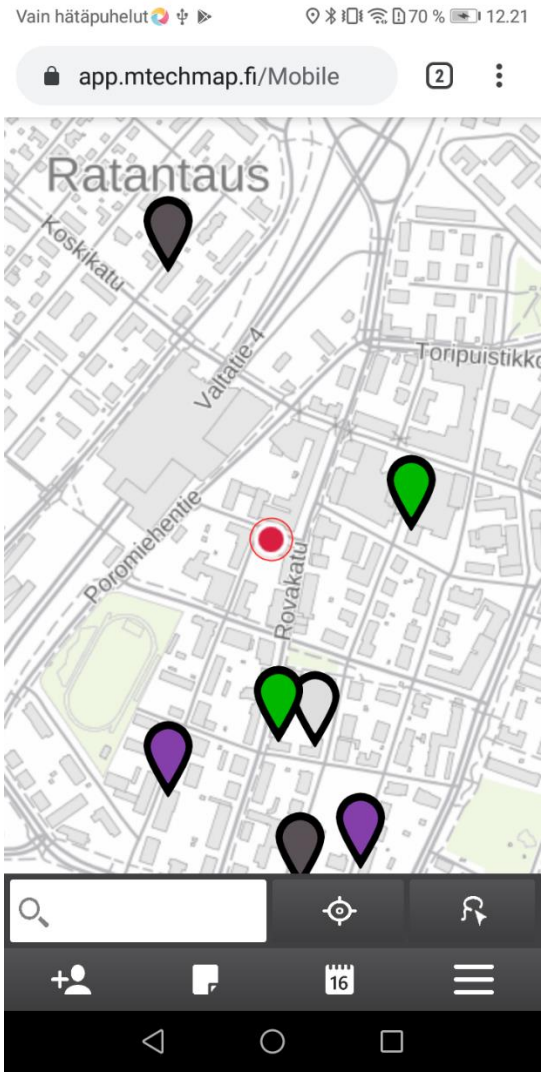
Testiajojen alustukseen testilaitteet pitää olla kytkettynä tietokoneeseen USB-johdolla ja Appiumin pitää olla päällä. Komentokehotteesta voidaan tarkistaa kaiken toimivan kuten pitääkin, kun testilaitteet on kytketty tietokoneeseen.

3.2.1 Honor 7A

Testilaitteen ollessa kytkettynä tietokoneeseen ja tarvittavien yhteyksien ollessa päällä, komentokehotteella avattiin adb-shell, jonka kautta pystyttiin seuraamaan puhelimen sovelluspaketteja ja niiden aktiviteettia. Aluksi testikoodin kirjoittaminen verkkoselainpohjaiselle sovellukselle oli haastavaa, sillä painikkeiden polut

oli vaikea löytää. Tässä vaiheessa en ollut vielä löytänyt Appiumin ominaisuutta, jolla tuodaan puhelimen näyttö tietokoneelle, ja jolla pystyy seuraamaan painikkeiden polkua. Tämä ominaisuus osoittautui erittäin hyödylliseksi myöhemmin. Tässä vaiheessa painikkeiden tunnisteet (ID:t) löytyivät, kun testi kaadettiin tahallisesti, palauttaen verkkosivun koodi, josta tiedot löytyivät manuaalisesti etsimällä. Näin testikoodista saatiin tehtyä toimiva versio Honor 7A:lle. Testikoodia kirjoittaessa koodin toimivuutta testattiin jatkuvasti testilaitteella. Koodia hiottiin niin kauan, että testitulos oli halutun kaltainen.

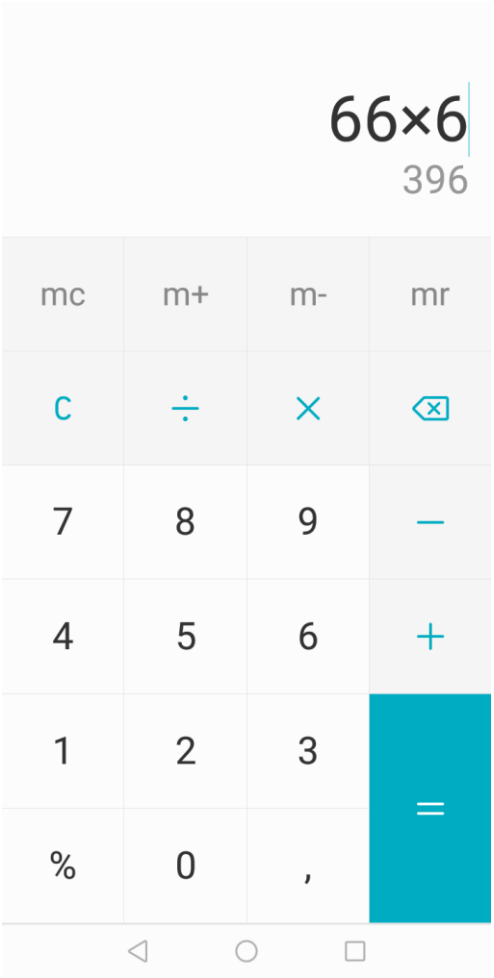
Automaatiotestissä testattiin MtechMap -sovellukseen sisäänkirjautumista. Kuviossa 4 on näkymä onnistuneesta kirjautumisesta testissä käytettyyn sovellukseen. Kuva on robotin testiajon aikana ottama kuvakaappaus näytöstä.



Kuvio 4. Näyttö robotin ajon aikana

Seuraavaksi testattiin matkapuhelimen laskinsovelluksen yleisominaisuuksia. Testissä käytettiin kertolaskuja, jakolaskuja sekä yhteen- ja miinuslaskuja. Testikoodin kirjoittaminen testilaitteen sovellukselle oli paljon yksinkertaisempaa ja helpompaa, sillä painikkeiden polut olivat nopeasti löydettävissä Appiumin avulla. Mobiililaitteen oman sovelluksen painikkeiden tunnisteet ovat muutenkin paljon yksinkertaisempia kuin verkkoselainpohjaisessa sovelluksessa.

Kuviossa 5 on testin viimeinen vaihe, kertolasku. Viimeisen onnistuneen laskutehtävän jälkeen sovellus ohjelmoitiin sulkemaan itsensä automaattisesti, minkä jälkeen testi ajaa itsensä alas.



Kuvio 5. Kertolasku Honor 7A

Kuviossa 6 nähdään robotin antama loppuraportti Ride-sovelluksen kautta.

```
elapsed time: 0:00:40    pass: 1    fail: 0
command: robot --argumentfile C:\Users\TUOMAS~1.JOK\AppData\Local\Temp\RIDEs2rrvtyt.d\argfile.txt
TestRunnerAgent: Running under CPython 3.7.2

==============================================================================
RobotFramework
==============================================================================
RobotFramework.CalculatorAndroid
==============================================================================
MtechMap base test                                                    | PASS |
------------------------------------------------------------------------------
RobotFramework.CalculatorAndroid                                      | PASS |
1 critical test, 1 passed, 0 failed
1 test total, 1 passed, 0 failed
==============================================================================
RobotFramework                                                        | PASS |
1 critical test, 1 passed, 0 failed
1 test total, 1 passed, 0 failed
==============================================================================
Output:  C:\Users\TUOMAS~1.JOK\AppData\Local\Temp\RIDEs2rrvtyt.d\output.xml
Log:     C:\Users\TUOMAS~1.JOK\AppData\Local\Temp\RIDEs2rrvtyt.d\log.html
Report:  C:\Users\TUOMAS~1.JOK\AppData\Local\Temp\RIDEs2rrvtyt.d\report.html

test finished 20191216 12:25:59
```

Kuvio 6. Testin tulos Honor 7A

Kuviossa 7 nähdään tarkka järjestyksen, missä painallukset tapahtuvat ja mitä robotti on tehnyt missäkin vaiheessa.

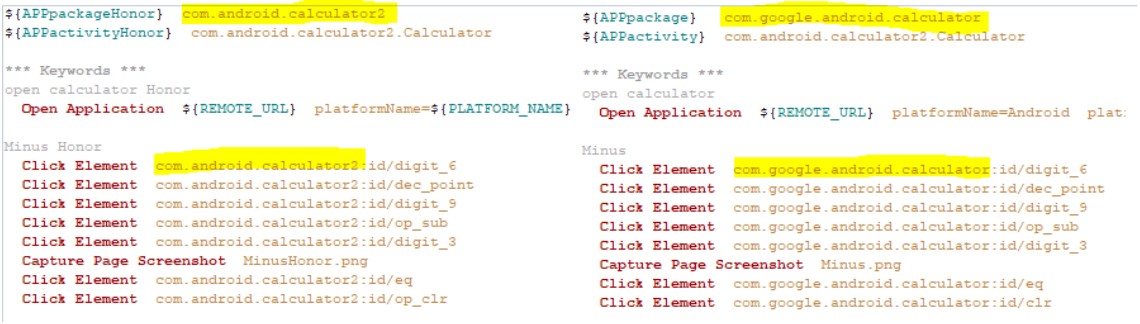
```
Starting test: RobotFramework.CalculatorAndroid.MtechMap base test
20191216 12:25:39.017 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:39.641 :  INFO : Clicking element 'com.android.calculator2:id/dec_point'.
20191216 12:25:40.354 :  INFO : Clicking element 'com.android.calculator2:id/digit_9'.
20191216 12:25:41.095 :  INFO : Clicking element 'com.android.calculator2:id/op_sub'.
20191216 12:25:41.784 :  INFO : Clicking element 'com.android.calculator2:id/digit_3'.
20191216 12:25:43.057 :  INFO : </td></tr><tr><td colspan="3"><a href="Minus.png"><img src="Minus.png" width="800px"></a>
20191216 12:25:43.059 :  INFO : Clicking element 'com.android.calculator2:id/eq'.
20191216 12:25:43.261 :  INFO : Clicking element 'com.android.calculator2:id/op_clr'.
20191216 12:25:44.475 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:45.202 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:45.904 :  INFO : Clicking element 'com.android.calculator2:id/op_add'.
20191216 12:25:46.579 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:47.818 :  INFO : </td></tr><tr><td colspan="3"><a href="Plus.png"><img src="Plus.png" width="800px"></a>
20191216 12:25:47.823 :  INFO : Clicking element 'com.android.calculator2:id/eq'.
20191216 12:25:47.980 :  INFO : Clicking element 'com.android.calculator2:id/op_clr'.
20191216 12:25:49.187 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:49.921 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:50.572 :  INFO : Clicking element 'com.android.calculator2:id/op_div'.
20191216 12:25:51.223 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:52.442 :  INFO : </td></tr><tr><td colspan="3"><a href="Divide.png"><img src="Divide.png" width="800px"></a>
20191216 12:25:52.444 :  INFO : Clicking element 'com.android.calculator2:id/eq'.
20191216 12:25:52.632 :  INFO : Clicking element 'com.android.calculator2:id/op_clr'.
20191216 12:25:53.819 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:54.550 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:55.203 :  INFO : Clicking element 'com.android.calculator2:id/op_mul'.
20191216 12:25:55.848 :  INFO : Clicking element 'com.android.calculator2:id/digit_6'.
20191216 12:25:57.085 :  INFO : </td></tr><tr><td colspan="3"><a href="Times.png"><img src="Times.png" width="800px"></a>
20191216 12:25:57.088 :  INFO : Clicking element 'com.android.calculator2:id/eq'.
20191216 12:25:57.290 :  INFO : Clicking element 'com.android.calculator2:id/op_clr'.
Ending test:   RobotFramework.CalculatorAndroid.MtechMap base test
```

Kuvio 7. Loki ajosta Honor 7A

Honor 7A määritettiin oletuslaitteeksi, ja laitteen tulokset toimivat vertailupohjana muiden laitteiden testituloksille. Kaikki testit siis kirjoitettiin ensimmäiseksi tälle laitteelle. Kun kaikki testiajot ja -koodit saatiin toimimaan oikein Honor 7A:lla, siirryttiin testaamaan muita laitteita samoilla koodeilla.

3.2.2 Nokia 5

Peruslaskintestissä Nokia 5 käyttää eri sovelluspakettia verrattuna Honor 7A:han, ja sen takia painikkeiden tunnisteet ovat myös erilaiset. Tämän vuoksi pääosa alkuperäisestä testikoodista jouduttiin muuttamaan niin, että tunnisteet olivat oikein (Kuvio 8).



```
${APPpackageHonor}   com.android.calculator2
${APPactivityHonor}  com.android.calculator2.Calculator

*** Keywords ***
open calculator Honor
  Open Application  ${REMOTE_URL}  platformName=${PLATFORM_NAME}

Minus Honor
  Click Element  com.android.calculator2:id/digit_6
  Click Element  com.android.calculator2:id/dec_point
  Click Element  com.android.calculator2:id/digit_9
  Click Element  com.android.calculator2:id/op_sub
  Click Element  com.android.calculator2:id/digit_3
  Capture Page Screenshot  MinusHonor.png
  Click Element  com.android.calculator2:id/eq
  Click Element  com.android.calculator2:id/op_clr
```

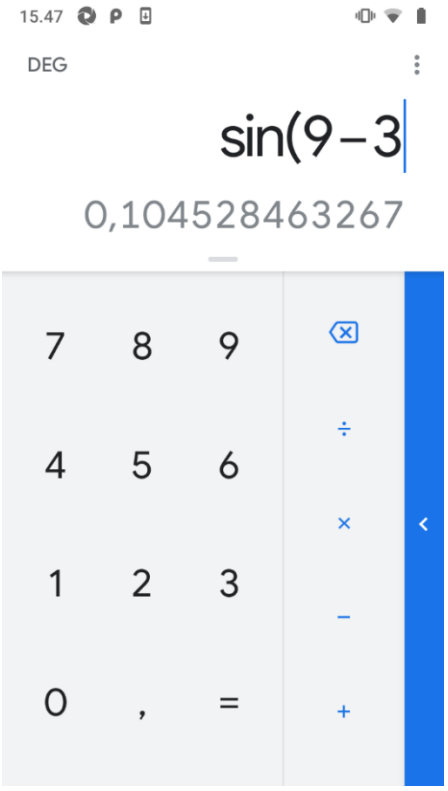
```
${APPpackage}   com.google.android.calculator
${APPactivity}  com.android.calculator2.Calculator

*** Keywords ***
open calculator
  Open Application  ${REMOTE_URL}  platformName=Android  plat

Minus
  Click Element  com.google.android.calculator:id/digit_6
  Click Element  com.google.android.calculator:id/dec_point
  Click Element  com.google.android.calculator:id/digit_9
  Click Element  com.google.android.calculator:id/op_sub
  Click Element  com.google.android.calculator:id/digit_3
  Capture Page Screenshot  Minus.png
  Click Element  com.google.android.calculator:id/eq
  Click Element  com.google.android.calculator:id/clr
```

Kuvio 8. Kahden koodin tunniste-erot

Vaikka testikoodin tunnisteet muokattiin Nokia 5:lle, tapahtui testissä kuviossa 9 näkyvä virhe. Tässä vaiheessa laskinsovellusten ero oli selvästi näkyvissä, sillä Nokia 5:llä alkunäkymä on eri kuin Honor 7A:lla. Honor 7A:lla näkymä avaa suoraan laskun 6,9 – 3, kun taas Nokia 5:llä numero 6 tilalle tuleekin sin(. Testin erot voi huomata parhaiten liitteenä olevasta videosta (Liite 2).



Kuvio 9. Miinuslasku Nokia 5

Nokia 5:llä verkkoselainsovelluksen testaus ei myöskään toiminut aluksi samalla testikoodilla kuin Honor 7A.lle. Näin ollen alkuperäistä testikoodia piti muokata, Nokia 5:n verkkoselain piti päivittää sekä asetuksia muuttaa täsmäämään Honor 7A:n verkkoselaimen asetuksia (kieli, selainversio ja verkkoselaimen etusivu). Näiden muokkausten jälkeenkin yhden elementin klikkaus toimii eri tavalla Nokia 5:llä kuin Honor 7A:lla. Alla näkyvässä kuviossa 10 näkyy, että käsky ”Go to url” on muutettu muotoon ”Input text”, sillä Nokia 5 ei muuten ymmärtänyt käskyä. Muilta osin testikoodi toimi oikein.

```
go to project web page
  Run Keyword If  "${DEVICE_NAME}" == "Nokia 5"  Run Keywords  Click Element  com.android.chrome:id/url_bar
  ...  AND  Input text  com.android.chrome:id/url_bar  https://app.mtechmap.fi/
  ...  AND  Nokia 5 steps
  Run Keyword IF  "${DEVICE_NAME}" == "Honor 7A"  Run Keywords  Click Text  Hae tai kirjoita verkko-osoite
  ...  AND  Go to url  https://app.mtechmap.fi
  ...  AND  Honor 7A steps

Nokia 5 steps
  @{GotoPageInputs}=  Get Webelements  class=android.widget.TextView
  Click Element  @{GotoPageInputs}[1]
  Sleep  3
  Click element  xpath=${basexpath}android.view.View[6]
  Sleep  5

Honor 7A steps
  Sleep  3
  Click element  xpath=${basexpathHonor}android.view.View[6]
  Sleep  5

```

Kuvio 10. Koodin eroavaisuus Honor 7A - Nokia 5

3.2.3 Iphone SE ja Ipad

Automaatiotestausta oli mahdotonta tehdä Iphone SE:lle ja Ipadille laiterajoitusten takia. Iphonen ja Ipadin SDK:ita ei pysty lataamaan muulle kuin MacOSx-käyttöjärjestelmälle, joten käytössä olevalla laitekokoonpanolla näitä laitteita ei pystytty testaamaan. Yhteensopiva käyttöjärjestelmä vaatisi mahdollisuutta käyttää virtuaalikonetta, fyysistä Applen Mac laitetta tai niin sanottua ”hackintoshia”, jossa asennetaan MacOSx-käyttöjärjestelmä muulle kuin Applen laitteelle.

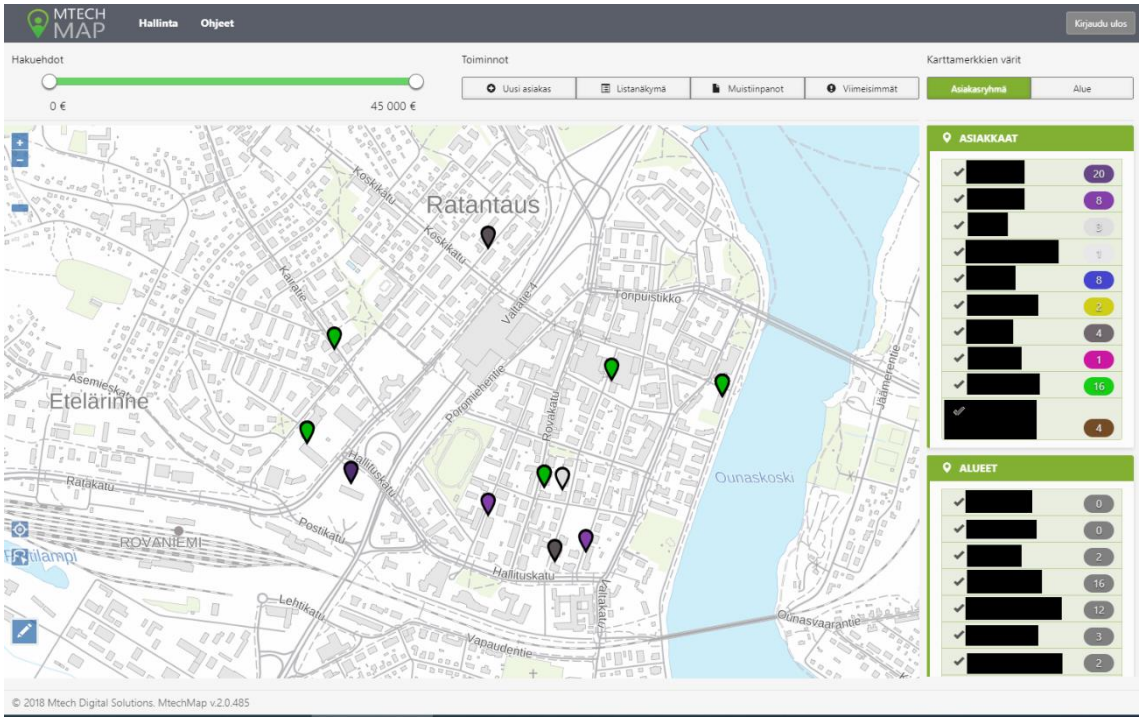
Myöskään Iphonea tai Ipadia ei voida asettaa kehittäjätilaan ilman MacOSx-käyttöjärjestelmää. Joten jouduin luopumaan ideasta, että voisin testata Iphonea tai Ipadia mitenkään tämänhetkisillä laitteilla.

3.2.4 Acer Iconia One 10 B3-A32 -tabletti

Laskinsovelluksen testauksessa Acerin Iconia One -tabletti toimi pääosin samoin kuin Honor 7A. Ainoastaan Tyhjennä-painikkeen tunniste oli eri Honor 7A:ssa kuin Acer Iconia Onessa. Honor 7A:ssa se on ”com.android.calculator2:id/op_clr” ja Acer Iconia Onessa pelkästään ”com.android.calculator2:id/clr”.

Verkkoselainsovelluksen testauksessa oli enemmän eroavaisuuksia Honor 7A:han verrattuna. Verkkoselainsovellukseen MtechMap sisäänkirjautuminen onnistui normaalisti, mutta sen jälkeen näkymä on erilainen kuin Honor 7A:ssa.

MtechMap-sovellus ei tunnista Acer Iconia One -laitetta mobiililaitteeksi, vaan käsittelee sitä tietokoneena. Tästä syystä sovellus näyttää täysin eriltä sisäänkirjautumisen jälkeen kuin Honor 7A:lla. Kuvio 11 on näyttökuva tietokoneelta, jota voi verrata Kuvioon 4 yllä.



Kuvio 11. Sisäänkirjautunut Acer Iconia One

3.2.5 Huawei Tab 7

Huawei Tab 7 –tabletilla laskinsovelluksen testauksessa painikkeiden tunnisteet olivat erilaiset verrattuna Honor 7A:han. Tunnisteiden eroavaisuudet näkyvät kuviossa 12. Esimerkiksi Huawein tabletilla desimaalipilkun tunniste on ”dot”, kun taas Honor 7A:lla se on ”dec_point”. Näin ollen testikoodia piti muokata niin, että tunnisteet olivat oikein. Tunnisteiden muutosten jälkeen laskintesti toimi oikein.

```
Minus Huawei
  Click Element  com.android.calculator2:id/digit6
  Click Element  com.android.calculator2:id/dot
  Click Element  com.android.calculator2:id/digit9
  Click Element  com.android.calculator2:id/minus
  Click Element  com.android.calculator2:id/digit3
  Capture Page Screenshot  MinusHuawei.png
  Click Element  com.android.calculator2:id/equal
  Sleep  1
  Click Element  com.android.calculator2:id/clear
```

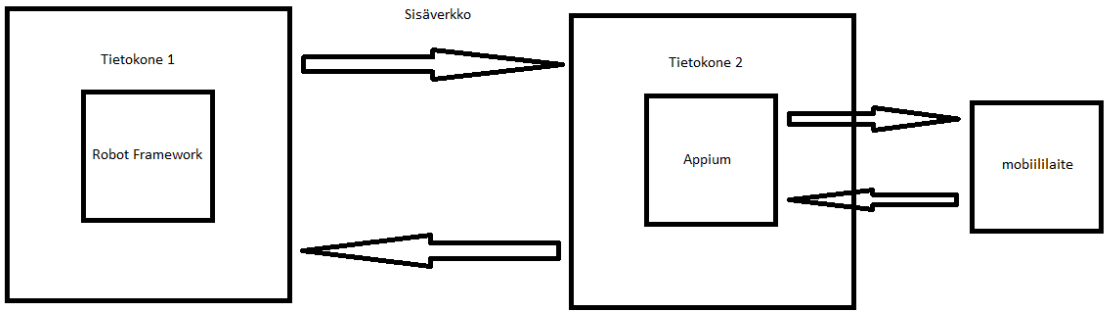
```
Minus Honor
  Click Element  com.android.calculator2:id/digit_6
  Click Element  com.android.calculator2:id/dec_point
  Click Element  com.android.calculator2:id/digit_9
  Click Element  com.android.calculator2:id/op_sub
  Click Element  com.android.calculator2:id/digit_3
  Capture Page Screenshot  MinusHonor.png
  Click Element  com.android.calculator2:id/eq
  Click Element  com.android.calculator2:id/op_clr
```

Kuvio 12. Koodi Huawei Tab 7 vertailtaessa Honor 7A:n

Verkkoselainsovelluksen testauksessa piti ensin tarkistaa, että selainversio oli päivitetty, ja että selaimen asetukset olivat samat kuin Honor 7A:ssa. Tämän jälkeen Honor 7A:lle tehty testikoodi toimi oikein ilman ongelmia.

3.2.6 Sisäverkon yli testaus toiselta tietokoneelta

Lopuksi testattiin vielä, toimiiko mobiililaitteiden testaus sisäverkon yli toisella tietokoneella. Testilaitteet yhdistettiin tietokoneeseen, jolla aiemmatkin testit tehtiin, mutta testiajot tehtiin samassa sisäverkossa olevalla toisella tietokoneella, johon myös asennettiin Robot Framework sekä Appium-kirjasto. Kuviossa 13 on havainnollistettu, miten laitteet on yhdistetty toisiinsa.



Kuvio 13. Tiedonkulku osien välillä sisäverkossa

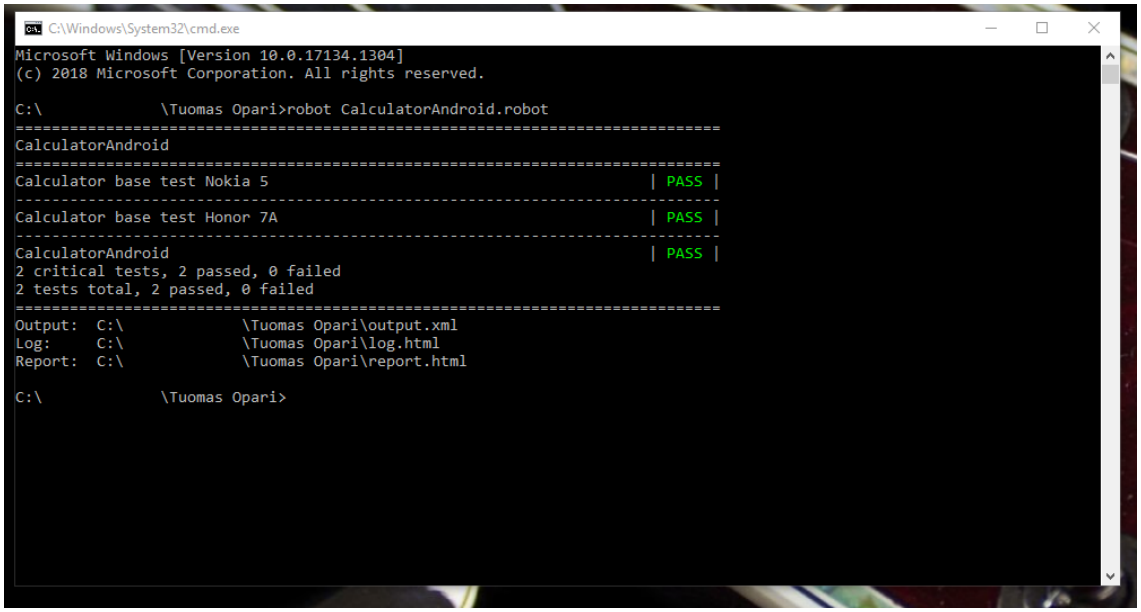
Sisäverkon yli testauksessa testikoodiin piti ainoastaan muokata Remote URL (Kuvio 14), joka on yllä olevassa kaaviossa (Kuvio 13) Tietokone 2:n URL.

```
*** Variables ***
${REMOTE_URL}    http://192.168.206.116:4723/wd/hub
${PLATFORM_NAME}  Android
${PLATFORM_VERSION}  9
${DEVICE_NAME}  Nokia 5
${APPpackage}    com.google.android.calculator
${APPactivity}   com.android.calculator2.Calculator
```

Kuvio 14. Muutettu URL

Alkuperäisessä testikoodissa Remote URL oli ”http://0.0.0.0:4723/wd/hub”, joka on Tietokone 2:n local URL. Tämän muutoksen jälkeen testiajot sisäverkon yli

toimivat oikein. Komentokehotteesta pystyy vielä tarkistamaan raportin testiajosta (Kuvio 15), josta näkee, että testi on ajettu onnistuneesti. Komentokehotteella onnistunut testi on merkitty vihreällä ”pass”, ja mikäli testi olisi epäonnistunut, testin kohdalla lukisi punaisella ”fail”.



Kuvio 15. Komentokehotteen raportti sisäverkon testiajosta

Sisäverkon yli tehtiin samat testit kuin aiemminkin, eli laskintesti sekä sisäänkirjautuminen MtechMap-sovellukseen. Testit tehtiin kahdelle eri laitteelle, Honor 7A:lle ja Nokia 5:lle.

4 POHDINTA

Opinnäytetyötä aloittaessani Robot Framework ja Ride olivat minulle tuttuja työkaluja jo verkkosivujen automaatiotestauksessa sekä rpa:ssa. Opinnäytetyössä oli lähtökohtana, että automaatiotestausta pystyy tekemään mobiililaitteille myös verkon yli, ja tämän vuoksi oikeiden työkalujen valintaa varten jouduin etsimään tietoa, mitkä työkalut tähän parhaiten soveltuvat. Lopulta työkaluiksi valittiin Robot Framework ja Appium. Appium olikin hyvä valinta, sillä Robot Frameworkille löytyy suoraan Appiumin kirjasto, josta löytyy hyvä dokumentaatiopohja ja tarpeeksi ominaisuuksia automaatiotestien tekemiseen mobiililaitteille ja mobiililaitteiden emulaatioille.

Robot Framework oli minulla jo valmiiksi käytettävissä, joten seuraavaksi asensin Appium-kirjaston Robot Frameworkille. Asennettuani Appiumin sovelluksen tietokoneelle piti lisäksi asentaa useampia Node:n kirjastoja sekä Androidin sdk:t, jotta Appiumin sai päälle. Tässä vaiheessa pystyttiin jo toteamaan, että Applen mobiililaitteiden testaus ei ole mahdollista käytettävissä olevilla laitteilla.

Testikoodin kirjoittaminen ensimmäiselle testilaitteelle oli suhteellisen yksinkertaista ja automaatiotestit onnistuivat hyvin. Haastetta toi kuitenkin se, että painikkeiden tunnisteiden löytämiseksi verkkoselainpohjainen sovellus piti tahallisesti kaataa, mikä oli erittäin aikaa vievää. Saman koodin käyttäminen muilla mobiililaitteilla ei onnistunut suoraan, vaan koodia piti jonkin verran muokata useimmille laitteille. Tässä vaiheessa olin löytänyt kuitenkin Appiumin ominaisuuden, joka yhdistää mobiililaitteen Appiumiin, näyttäen mobiililaitteen näytön tietokoneella antaen lisätietoa laitteen painikkeista.

Automaatiotestit eri mobiililaitteilla ajettiin onnistuneesti läpi muokatuilla testikoodeilla. Lähtökohtaisesti toiveena oli, että sama testikoodi olisi toiminut sellaisenaan kaikille eri mobiililaitteille, mikä ei näin ollen toteutunut. Suurin osa testeissä huomatuista eroavaisuuksista löytyi laskintesteissä. Niissä painikkeiden tunnisteet ja sovelluspaketit olivat erilaisia. Verkkoselainsovellusta testatessa Nokia 5 -matkapuhelimen verkkoselaimen osoitepalkin toiminnallisuus oli erilainen kuin Honor 7A:ssa. Verkkoselainsovellus ei myöskään tunnistanut Acerin tablettia mobiililaitteeksi, vaan käsitteli sitä tietokoneena, jonka vuoksi sisäänkirjautumisnäkymä oli erilainen.

Appium mahdollisti sisäverkon yli testauksen helposti, sillä testikoodia ei tarvinnut tässä vaiheessa enää muokata muutoin kuin url:n osalta. Appiumin osalta ei tarvinnut muokata mitään, vaan se toimi valmiiksi sisäverkon IP-osoitteessa. Näin ollen oli riittävää, että testit tehtiin vain kahdella eri mobiililaitteella onnistuneesti. Tästä voidaan päätellä, että url:n muutoksella myös muiden mobiililaitteiden testikoodit olisivat toimineet. Jatkoa ajatellen, Appium voisi mahdollistaa mobiililaitteiden käytön myös etänä niin, että kaikilla samassa sisäverkossa olevilla olisi mahdollisuus käyttää samoja testilaitteita.

Loppupäätelmänä voi sanoa, että mobiililaitteiden testaus yksi kerrallaan oli onnistunut. Kuitenkaan useamman mobiililaitteen testaus samalla koodilla ei ole mahdollista, sillä mobiililaitteiden alkuperäissovellukset ovat erilaisia ja käyttävät valmistajasta riippuen eri versiota käyttöjärjestelmästä. Näiden tulosten perusteella en koe järkeväksi verkkosivupohjaisen sovelluksen automaatiotestausta useammalla mobiililaitteella, sillä testikoodin ylläpitäminen olisi raskasta ja aikaa vievää mobiililaitteiden päivittyessä.

Jatkossa tällaisen automaatiotestauksen tekeminen voisi olla järkevää, kun tehdään mobiilisovellusta. Tällöin voitaisiin olettaa, että sovelluspaketti on samanlainen mobiililaitteesta riippumatta, ja painikkeiden tunnisteet ovat samoja, ja näin ollen testikoodia on helpompi ylläpitää.

LÄHTEET

Appium 2020. Introduction. Viitattu 23.4.2020
http://appium.io/docs/en/about-appium/intro/?lang=en.

AppiumLibrary 2020. Viitattu 23.4.2020
https://serhatbolsu.github.io/robotframework-appiumlibrary/AppiumLibrary.html.

Atrsoft 2018. Testausautomaatio tehostaa laadunvarmistusta. Viitattu 24.4.2020
https://www.atrsoft.com/2018/02/12/testausautomaatio-tehostaa-laadunvarmistusta/.

Lepistö, J. 2020. Testiautomaation maailma: Robot Framework ja Appium. Viitattu 23.4.2020
http://www.qubilea.fi/testiautomaation-maailma-robot-framework-ja-appium/.

Mtech Digital Solutions Oy 2020. Tietoa meistä. Viitattu 21.4.2020
https://www.mtech.fi/tietoa-meista/.

Robot Framework 2020. Introduction. Viitattu 23.4.2020
https://robotframework.org/.

LIITTEET

Liite 1. Kalenterin avaus, taulukko

Liite 2. Calculator2Devices, video

Liite 1. Kalenterin avaus, taulukko

Vaatimus	Testi	Esiehdot	Odotettu Lopputulos
Kalenteri ikonia painamalla avautuu kalenteri, jossa KALENTERI, MUISTIO ja LOKI välilehdet.	Kartta näkymässä alapalkista kalenteri painiketta painetaan	Asiakas on kirjautunut sisään	Kalenteri näkymä avautuu
Kalenteri näkymässä voit valita päivän ja nähdä kyseisen päivän tapahtumat	valitset näytöltä päivämäärän		Päivämäärän tapahtumat näkyvät kalenterin alla
Kalenterinäkymässä on kuukausikalenteri (Vasemmalle/ oikealle kalenterin kohdalla siirtyy edelliseen/seuraavaan kuukauteen)	Painat nuolta sivulle joka vaihtaa kalenteri näkymän kuukauden joko seuraavaan tai edelliseen kuukauteen		Kuukausi vaihtuu kalenteri näkymässä
Vuoden ja kuukauden vaihtaminen näkymässä vetovalikkoa käyttäen	Painan veto valikkoa kuukauden tai vuoden kohdalla.		Veto valikko avautuu ja pystyt valitsemaan kuukauden tai vuoden
Kalenterin pystyy sulkemaan	Painan Kalenteri näkymän "X" painiketta	Kalenteri näkymä on avattuna	Kalenteri näkymä sulkeutuu

Liite 2. Calculator2Devices, video

https://www.youtube.com/watch?v=whMEAJHrb40